Probing Methods for Generalized Saddle-Point Problems

Chris Siefert and Eric de Sturler

Abstract

Several Schur complement-based preconditioners have been proposed for solving (generalized) saddle-point problems. We consider probing-based methods for approximating those Schur complements in the preconditioners of the type proposed by [Murphy, Golub and Wathen '00], [de Sturler and Liesen '03] and [Siefert and de Sturler '04]. This approach can be applied in similar preconditioners as well. We discuss the implementation of probing-based approximations to Schur complements. We consider the application of those approximations in preconditioners for Navier-Stokes problems and metal deformation problems. Finally, we present eigenvalue clustering for the preconditioned matrices, and convergence and timing results. These demonstrate the effectiveness of the proposed preconditioners with probing-based approximate Schur complements.

1 Introduction

Generalized saddle-point problems [23] are of the form,

$$\mathcal{A} \left[\begin{array}{c} x \\ y \end{array} \right] \equiv \left[\begin{array}{cc} A & B^T \\ C & D \end{array} \right] \left[\begin{array}{c} x \\ y \end{array} \right] = \left[\begin{array}{c} f \\ g \end{array} \right], \tag{1}$$

where $A \in \mathbb{R}^{n \times n}$, $D \in \mathbb{R}^{m \times m}$, and here $n > m$. For some problems, especially those arising in constrained optimization, $D = 0$. For others, such as those arising from stabilized finite elements [2, 13, 25], $D \neq 0$, but $\|D\|$ is small. For still others, the non-zero D arises from another source, such as a very slight compressibility in metal deformation problems [30]. For all the problems we consider $D = 0$ or $\|D\|$ is small, so that these problems retain the character of a generalized saddle-point problem. In addition, certain finite element stabilization schemes yield $B \neq C$ [1, 23] and [25, Sections 7.5 and 9.4], while many other schemes yield $B = C$. We consider the problem and preconditioners in the generalized form, $B \neq C$, mainly to emphasize the general applicability of the proposed methods. However, most of the paper is not specific to the generalized problem, and there is no particular emphasis on the generalized problem. A number of preconditioners for this class of problems have been developed that employ a Schur complement [3, 10, 11, 19, 22], or some approximation thereof [12, 14, 20, 24, 28]. Section 2 will present a summary of the convergence results for one such family of preconditioners [28].

For some problems, there is an obvious approximation to the Schur complement arising in these preconditioners that yields good convergence, like the pressure mass matrix for the Navier-Stokes problem [29]. For other problems, however, no obvious approximation exists. Probing [4, 18] provides a general, algebraic method for building matrix approximations. It was designed to construct narrowly-banded approximations to matrices arising in 2-D domain decomposition [4], but we require approximations with a more general sparsity pattern. Graph coloring techniques used by the optimization community for sparse Jacobian and Hessian approximations [6, 7, 16, 17, 21] have been recently adapted to allow the approximation of any matrix, so long as the sparsity pattern is known *a priori* [8].

We propose to apply these techniques for approximating Schur complements. A potential problem is that the matrices formed by these more general probing techniques can be expensive to factor. However, in previous experiments for exact Schur complements we found that using an incomplete factorization has a negligible effect on convergence. Therefore, we propose to use incomplete factorizations for these probing-based approximate Schur complements. This will keep the total cost of constructing and applying the approximate Schur complement linear in m. Finally, we note that any matrix that has entries which decay with distance on some underlying graph, e.g. a graph derived from the underlying finite element mesh, is a candidate for these more advanced probing methods, and this relationship will be explained in more detail in Section 3. Algorithmic and implementation details will be the focus of Sections 4, 5 and 6.

Approximating Schur complements by probing with general sparsity patterns combined with incomplete factorizations for the resulting approximation gives cheap and effective preconditioners with excellent performance. We demonstrate our preconditioners for two applications, the first in fluid flow and the second in metal deformation. Analysis of the eigenvalues of the preconditioned systems, as well as GMRES convergence results and timings will be provided in Section 7. Section 8 summarizes our conclusions and presents directions for future work.

2   Preconditioning Saddle-Point Problems

In [28], we developed two classes of preconditioners for (1) and provided eigenvalue bounds for the corresponding preconditioned systems that allow for the use of approximate Schur complements. Both classes of preconditioners require the choice of a splitting of the (1,1) block of (1), namely $A = F - E$, where F is cheap to solve with. Let $S_1 = -(D - CF^{-1}B^T)$ be the exact Schur complement for the preconditioner, and let S_2 be an approximation to S_1. Also, let $\mathcal{E} = S_2^{-1}S_1 - I$. If we precondition from the left, the system looks as follows,

$$\begin{bmatrix} F^{-1} & 0 \\ 0 & S_2^{-1} \end{bmatrix}\begin{bmatrix} A & B^T \\ C & D \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} I-S & N \\ M & Q \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \tilde{f} \\ \tilde{g} \end{bmatrix}, \tag{2}$$

where $S = F^{-1}E$, $N = F^{-1}B^T$, $M = S_2^{-1}C$ and $Q = S_2^{-1}D$. We will refer to this as the block-diagonally preconditioned system. The second preconditioned system, which will be referred to as the *related system*, is derived from a further splitting of the block-diagonally preconditioned system as follows [28],

$$\begin{bmatrix} I-S & N \\ M & Q \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \left(\begin{bmatrix} I & N \\ M & MN-I \end{bmatrix} - \begin{bmatrix} S & 0 \\ 0 & \mathcal{E} \end{bmatrix}\right)\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \tilde{f} \\ \tilde{g} \end{bmatrix}. \tag{3}$$

Note that,

$$\begin{bmatrix} I & N \\ M & MN-I \end{bmatrix}^{-1} = \begin{bmatrix} I-NM & N \\ M & -I \end{bmatrix}. \tag{4}$$

Multiplying (3) by (4) yields

$$\begin{bmatrix} I-(I-NM)S & -N\mathcal{E} \\ -MS & I+\mathcal{E} \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \hat{f} \\ \hat{g} \end{bmatrix}. \tag{5}$$

This is the related system for the fixed-point iteration derived from the splitting in (3). Multiplying with this matrix is more expensive than with the block-diagonally preconditioned matrix. It requires two additional

applications of F^{-1}, one additional multiplication with C and two with B^T. However, the greatly improved eigenvalue clustering of (5) makes the second method much cheaper in practice, and in general we recommend solving the related system over solving the block-diagonally preconditioned system.

In [28], we also provide bounds that describe the eigenvalue clustering of the preconditioned systems proposed. Eigenvalue clustering is a key factor in the convergence of Krylov subspace methods, although in non-symmetric problems the eigenvector matrix also plays a role. Tight clustering of the eigenvalues generally leads to fast convergence. We now briefly summarize these eigenvalue bounds.

For the related system (5), the eigenvalues are clustered around 1, and we have the following bound [28, Theorem 4.2]. Let λ_R be an eigenvalue of the matrix in (5). Then

$$|\lambda_R - 1| \leq \sqrt{1 + \|N\|_2^2}\sqrt{1 + \|M\|_2^2}\max\left(\|S\|_2, \|\mathcal{E}\|_2\right).$$

We have several bounds for the block-diagonally preconditioned system (2). Let $\hat{Q} = S_1^{-1}D$, with $\hat{Q}V = V\Delta$, $\Delta = \mathrm{diag}(\delta_j)$, and $\delta_j \neq -1$ for any j. Furthermore, let ω_1 be the cosine of the smallest canonical angle between range (NM) and null (NM), and let Θ be such that $NV\Theta^{-1}$ has orthonormal columns. We refer to [28] for details. Finally, let $\lambda_{S,\mathcal{E}}$ be an eigenvalue of the preconditioned matrix in (2). Then, depending on the values of the δ_j, one of the following bounds on $|\lambda_{S,\mathcal{E}} - \lambda|$ holds for some $\lambda \in \left\{1, \frac{1+\delta_j \pm \sqrt{4+(1+\delta_j)^2}}{2}\right\}$ [28, Theorem 4.1].

If $D = 0$, we have

$$|\lambda_{S,\mathcal{E}} - \lambda| \leq 2\left(\frac{1+\omega_1}{1-\omega_1}\right)^{-1/2}\|S\|_2 + \frac{2\sqrt{5}}{5}\|\mathcal{E}\|_2.$$

If the δ_j's are real, we have

$$|\lambda_{S,\mathcal{E}} - \lambda| \leq a_1(\Theta)\left(\frac{1+\omega_1}{1-\omega_1}\right)^{-1/2}\|S\|_2 + a_2(\Delta, V)\|\mathcal{E}\|_2.$$

If δ_j's are complex and $\exists \alpha > 0$ s.t. $|\delta_j| \leq \alpha < \sqrt{5}$, we have

$$|\lambda_{S,\mathcal{E}} - \lambda| \leq a_3(\Theta, \alpha)\left(\frac{1+\omega_1}{1-\omega_1}\right)^{-1/2}\|S\|_2 + a_4(\alpha, V)\|\mathcal{E}\|_2.$$

The functions a_1, a_2, a_3 and a_4 are discussed in detail in [28]. Here, it is important that these functions do not depend on $\mathcal{E}$ (explicitly or implicitly). Furthermore, they are large only if Θ or V is ill-conditioned, or if some δ_j is near $\sqrt{5}$.

These bounds for the eigenvalues of the related system (5) and the block-diagonally preconditioned system (2) indicate that the perturbation of the eigenvalues from those of the corresponding preconditioner with the exact Schur complement is at most linear in $\|\mathcal{E}\|_2$.

3 Probing

The probing method [4] was developed to approximate Schur complements arising from the interfaces in domain decomposition problems without explicitly forming the matrices. The method approximates a matrix

using only matrix-vector multiplication. It multiplies a few carefully constructed vectors by the matrix and then constructs the approximate matrix from the results, based on a sparsity pattern chosen *a priori*.

"Classic" probing (as introduced by Chan and Mathew [4]) assumes that the matrix is banded with bandwidth b, and approximates the matrix based on that assumption. If this assumption is true, then probing is exact. If not, then probing yields a banded approximation and lumps entries of the matrix outside the band into the band of the approximate matrix. For example, consider the tridiagonal matrix shown in Figure 1. We choose $e_1 + e_4$, $e_2 + e_5$ and e_3 as our probing vectors and we multiply these by the matrix. The resulting product recovers all the non-zero entries in the original matrix exactly.

$$
\begin{bmatrix}
a_1 & b_2 & & & \\
c_1 & a_2 & b_3 & & \\
 & c_2 & a_3 & b_4 & \\
 & & c_3 & a_4 & b_5 \\
 & & & c_4 & a_5
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
1 & 0 & 0 \\
0 & 1 & 0
\end{bmatrix}
=
\begin{bmatrix}
a_1 & b_2 & 0 \\
c_1 & a_2 & b_3 \\
b_4 & c_2 & a_3 \\
a_4 & b_5 & c_3 \\
c_4 & a_5 & 0
\end{bmatrix}
$$

Figure 1. Classic probing on a tridiagonal matrix, using the vectors $e_1 + e_4$, $e_2 + e_5$ and e_3.

Classic probing is cheap to implement. Assume that the matrix we are approximating is $n \times n$ and we are using b vectors, i.e., our *a priori* sparsity pattern is chosen to construct an approximation with bandwidth b. Computing the probing vectors and building the approximate matrix cost $O(nb)$ computational work. However, performing the matrix-vector multiplications will be more expensive. If the matrix we are trying to approximate has bandwidth b (the case of exact reconstruction), the matrix-vector multiplications cost $O(nb^2)$ work. If the matrix is dense or has a larger bandwidth, then the matrix-vector multiplications will be even more expensive.

Probing works well for the application in [4], since the entries of the exact Schur complement decay like $O(|i-j|^{-2})$. Thus, a banded approximation computed by probing approximates the large entries in the Schur complement accurately and yields a good overall approximation. We propose to use probing for more complicated matrices, specifically matrices whose coefficients decay in magnitude as a function of distance on some underlying graph. Consider the problem

$$
-\Delta u + .25u = 0 \tag{6}
$$

in two dimensions with homogeneous Dirchlet boundary conditions. We discretize the problem with finite differences, using 15 grid points in each dimension. Figure 2 shows one column of the inverse of the matrix from (6) shown over the finite difference grid. Note that the inverse of the matrix from (6) has entries (i, j) that rapidly decay in magnitude as the distance between nodes i and j on the grid increases. Such a matrix can be approximated accurately with a probing-based method, but not by classic probing, unless a large number of vectors is used. The banded reconstruction used by classic probing will only capture decay in one direction (corresponding to neighbors that are "close" in the node numbering) unless a very large number of vectors is used. Thus, while classic probing can capture 1-D decay very well, for more complicated decay patterns, like the one resulting from (6), we need more advanced methods than the classic probing method [4]. This is even more the case for problems in three spatial dimensions and for systems of partial differential equations.

The partial matrix estimation technique [8], which has been used in the estimation of sparse Jacobians and Hessians under different names [6, 7, 16, 17, 21], provides a way for approximating more general matrices.
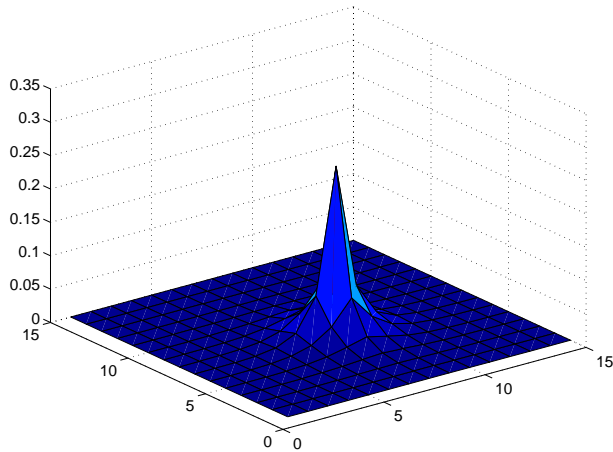
Figure 2. One column of the inverse of the matrix from (6) graphed over the underlying finite difference mesh.

Here, we shall refer to this method as *structured probing*. In this technique, we first choose a sparsity pattern for the approximate matrix, based on *a priori* knowledge of the matrix we are approximating. Second, we use graph coloring techniques to compute probing vectors such that a matrix of our chosen sparsity pattern would be reconstructed exactly (see Section 4). Third, we multiply the probing vectors by the matrix. Finally, we use the results of the matrix-vector multiplication to approximate the matrix according to our chosen sparsity pattern (see Section 5). Algorithm 1 outlines the process for a given input matrix K.

Algorithm 1 $\tilde{K} = \mathsf{Structured\ Probing}(K \in \mathrm{I\!R}^{n \times n})$

1: Choose a sparsity pattern (matrix) $H \in \{0,1\}^{n \times n}$ for the output matrix $\tilde{K}$.
2: Perform a graph coloring on a graph derived from the matrix H to generate the vector $d \in \{1, 2, \ldots, p\}^n$, where p is the number of colors used by the graph coloring (generally not fixed in advance). The color for vertex i is given by $d(i)$.
3: Generate p probing vectors, $x_1, \ldots, x_p$, one for each color.
4: Compute $w_i = K x_i$, for $i = 1, \ldots, p$.
5: Build $\tilde{K}$ using sparsity pattern H and vectors $w_1, \ldots, w_p$.

Choosing a good sparsity pattern, H, in Step 1 of Algorithm 1, requires *a priori* knowledge of the matrix K. If the "big" entries in K are in a certain pattern, we can choose the *a priori* sparsity pattern accordingly. For many applications these large entries will be related to locality on some graph. If the problem is derived from a finite element discretization, the finite element mesh or a subset thereof may provide such a graph. If such a graph is not available immediately from the problem, we can often derive a suitable graph from the matrices involved. For example, in our case we are interested in matrices of the type $K = (D - CF^{-1}B^T)$ where F is closely related to A in (1). If A is related to the discretization of an elliptic partial differential equation, we can expect the large entries in F^{-1} to be related to the adjacency graph of F or A (or of powers of these matrices), and we can expect the entries to decay with distance over such a graph. These same principles have been exploited successfully in the computation of sparse approximate inverse preconditioners [5]. If the matrices B, C, and D are also sparse, we can in general derive a graph over which the entries of

K will decay with distance. This will be the case especially if B and C correspond to local operators such as the divergence of a vector field and the gradient of a function. Should F not be available, which is the case if F^{-1} is approached directly, e.g., using multigrid, we can use the adjacency graph of $(D - CA^kB^T)$ (for some modest k) instead. Finally, if we know that our matrix is related to the discretization of some operator, we can use a reasonable stencil for that operator to form our sparsity pattern.

If we consider probing to approximate the inverse of the matrix that arises from discretizing the problem (6), the superiority of structured probing over classic probing is quite clear. Figure 3 shows a column of the inverse of the matrix that arises from discretizing (6) as well as the corresponding columns in the approximations generated by structured and classic probing. For structured probing, we assume a 13-pt stencil, which for our particular choice of graph coloring, takes 15 vectors. For proper comparison, we also use 15 vectors for classic probing. Note that structured probing approximates this inverse martix significantly better than classic probing.
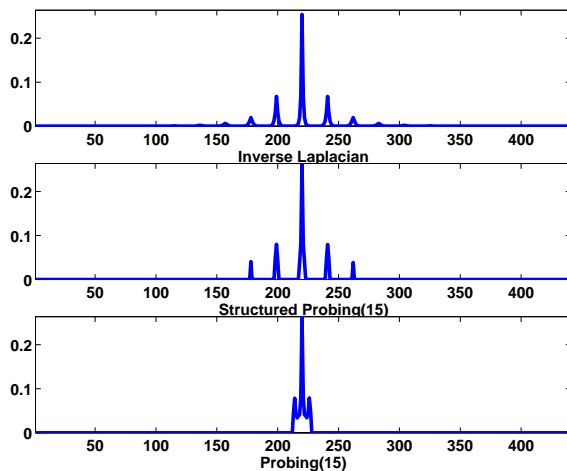

Figure 3. One column of the inverse of the matrix from (6) and the approximations generated by classic and structured probing with 15 vectors.

4   Graph Coloring

Graph coloring algorithms have long been used in computing sparse Jacobians [6, 7, 9, 16, 17, 21]. The standard graph coloring problem, also known as distance-1 coloring, is to assign colors to the vertices of a graph such that neighboring vertices have different colors. The simplest graph coloring algorithm, known as the *greedy* algorithm, considers the vertices sequentially and assigns each vertex the lowest numbered valid color.

In Step 2 of Algorithm 1, we generate a vector of colors, d, which we use in the construction of the probing vectors. This coloring *must* ensure that any single probing vector does not capture data from two columns with overlapping nonzero patterns. Methods known as *substitution* methods [8, 16] allow this condition to be relaxed, but they introduce other problems, and we will not consider them in this paper. The above condition is then a critical requirement for our probing vectors, and it raises two questions about the use of

graph coloring. The first question is which graph can we use to generate vectors that meet this requirement. The second question is how should we color this graph.

To describe our approaches, we consider the exact reconstruction of a sparse matrix K with at most b nonzero entries per row using a pattern $H \in \{0,1\}^{n \times n}$ such that $h_{i,j} = 1$ if and only if $k_{i,j} \neq 0$. The algorithms discussed below use different graphs, but they have the same vertex set, $V = \{v_1, \ldots, v_n\}$, corresponding to the n unknowns or n rows/columns of the matrix. For each algorithm, let p be the number of colors used.

From the perspective of efficiency, we would like these methods to have about the same cost as the necessary matrix-vector multiplications. For the problem of reconstructing K, the matrix-vector multiplications cost $O(nbp)$ [1]. For all problems, it is known that $p \geq b + 1$.

4.1 Choosing a Graph to Color

A natural graph to consider is $G_1(H) = (V, E_1)$, where $(v_i, v_j) \in E_1$ if and only if $h_{i,j} \neq 0$ or $h_{j,i} \neq 0$. This is commonly referred to as "the graph of the matrix," or the adjacency graph of the matrix H [21, 7]. A distance-1 coloring of G_1 does not guarantee that our condition on the probing vectors holds; however a distance-2 coloring does. In a distance-2 coloring no two neighbors, or neighbors of neighbors, have the same color [21]. If H is stored in a suitable sparse format, generating the graph is effectively free. Computing the distance-2 coloring takes $O(nb^2)$ work.

Another graph to consider is the *column intersection* graph, $G_2(H) = (V, E_2)$, where $(v_i, v_j) \in E_2$ if and only if there is a k such that $h_{k,i} \neq 0$ and $h_{k,j} \neq 0$ [6]. In the nomenclature of graph theory, these columns are not structurally orthogonal. For this graph, we can use a distance-1 graph coloring to ensure the aforementioned requirement on our probing vectors. This graph has $O(nb^2)$ edges, so it takes at least that much work to construct [17]. Our implementation [27] uses $O(nb^2 \log b)$ time to construct the graph in CSR format, and performing the coloring takes an additional $O(nb^2)$ time. If the graphs of both H and H^T are available, one can color the column intersection graph without forming it explicitly [6]. But in general, coloring the column intersection graph takes more work than doing the appropriate coloring on the adjacency graph [16].

A final option, which is better suited for rectangular matrices, is to consider a bipartite graph, where the vertex set V is partitioned into two disjoint subsets, V_1, which represents the rows of the matrix and V_2 which represents the columns [17]. This method requires a distance-2 partial coloring (the rows need not be colored) [17].

4.2 Choosing the Method to Color

After we choose the graph to color, we must choose a method to compute the coloring. The greedy algorithm is an obvious choice, but balanced coloring, as described in [8], may be a better option. In the latter approach, we balance the number of nodes assigned to each color. A simple heuristic for balanced coloring is to assign the least used valid color when multiple colors are valid for a node.

Another option is to use probing vectors with regular patterns (like those of [4]), but pick the number of vectors such that we can exactly reconstruct the desired sparsity pattern. One way to do this is to choose the number of vectors to be relatively prime to the differences between column indices of nonzero coefficients in a row, for all rows. More precisely, choose the number of colors p, such that p is relatively prime to all elements of the set $\{\ i - j \mid$ for some k, $h_{k,i} = 1$ and $h_{k,j} = 1\}$. We will refer to this method as the *prime divisor coloring*. As the method is based on the adjacency graph, we need a distance-2 coloring. This

---

[1] In practice, of course, K is only available implicitly through matrix-vector products. The cost then depends on how the required data is available and the implementation of the matrix-vector products.

approach often requires more colors than the other approaches. However, if H comes from a fixed stencil on a regular grid, we need only consider a single "representative" row of the matrix (i.e., a row for a point away from the boundaries), and use that row to choose the number of colors p. While this method takes $O(n(p + b^2))$ work for an unstructured mesh where b is the highest vertex degree, it takes only $O(p + b^2)$ work for problems on a regular grid.

As noted in [6], graph coloring heuristics are sensitive to the ordering of the nodes. A good ordering reduces the number of colors, and thus makes the underlying probing process computationally less expensive. The simplest such reordering is to order the nodes so that all of the high-degree vertices are numbered first. This is referred to as the largest-first ordering(LFO) in [6]. However, more complicated orderings, where nodes are colored based on the topology of particular subgraphs (such as the nodes numbered or unnumbered at the current stage of the algorithm) are discussed in [6].

Once we have a coloring, we can generate the probing vectors (Step 3 of Algorithm 1), perform the matrix-vector multiplications (Step 4) and construct the output matrix $\tilde{K}$ with the chosen sparsity pattern (Step 5). The matrix-vector multiplication depends on how K is represented, which in turn depends on the application. Next, we discuss the other two steps.

5 Building the Probing Vectors and Reconstructing the Matrix

Given the vector $d \in \{1, 2, \ldots, p\}^n$ that indicates the color for each vertex, we create the vectors $x_1, \ldots, x_p \in \{0, 1\}^n$ such that the j-th entry of x_i is equal to one if $d_j = i$ and zero otherwise. So, the j-th entry of x_i equals one if vertex j has the color i. Thus, the vector $w_i = Kx_i$ contains entries only from columns associated with vertices that have color i. By construction such columns do not overlap.

The reconstruction of the matrix is then a matter of using the sparsity pattern H to put the entries in the vectors $w_i = Kx_i$ at the right place in the output matrix $\tilde{K}$. Let $h_{k,j} = 1$. The vector x_{d_j} captures the j-th column of the matrix K. Therefore, the vector w_{d_j} gives the value for $\tilde{K}_{k,j}$ in its k-th entry; $\tilde{K}_{k,j} = w_{d_j}(k)$.

Figure 4 shows MATLAB-style pseudo-code for implementing these portions of Algorithm 1. This example is derived from the relevant routine in our C-language structured probing software library [27]. A more extensive C++ library (focused on sparse Jacobian and Hessians) is under development by Alex Pothen and his collaborators. In this example, we store the matrix in compressed sparse row (CSR) format [26, p. 90], with the entries in a row ordered by increasing column number. We assume that a graph p-coloring has been computed in advance and is given by the vector d.

6 Implementing Structured Probing

Implementing graph coloring algorithms efficiently can be complicated. The prime divisor and greedy methods discussed in Section 4 are straightforward to implement. However, the balanced coloring approach of [8] is more complicated and warrants further discussion.

The key difficulty in implementing balanced coloring [8] is efficiently finding the valid color that has been assigned to the smallest number of nodes. Since we need at least as many colors as our highest degree node plus one [17], we start the algorithm using that many colors. We increase the number of colors when a node cannot be assigned any of the existing colors. Algorithm 2 outlines a method for computing a balanced coloring on a graph.

Algorithm 2 $d = $ Balanced Coloring Heuristic(V, E)

1: Let n be the number of nodes in V. Let $p = 1 + \max \deg(V)$, the initial number of colors.
2: Let $c_i = 0$, for $i = 1, \ldots, p$, be the number of times each color has been used.
3: For $j = 1, \ldots, n$
4:     Let $S \subseteq \{1, \ldots, p\}$ be the set of colors that are valid for node j.
5:     **if** $S = \emptyset$ **then**
6:         Let $p = p + 1$, $k = p$ and $c_k = 0$ (use a new color).
7:     **else**
8:         Find $k = \arg\min_{l \in S} \{c_l\}$, the color to use.
9:     Let $d_j = k$ and $c_k = c_k + 1$ (assign color k to node j and update the color counts).

Efficiently computing S and k is key to an efficient implementation. We consider two approaches. Both rely on a list of colors sorted by the number of nodes with that color (c_i). The first approach is to consider each color and check all neighbors to see if that color is allowed (for distance-2 coloring this includes checking distance-2 neighbors). The second approach is to consider each neighbor and remove the invalid colors. Our implementation follows the second approach. We maintain a doubly linked list of colors sorted by the number of nodes with that color. To remove colors from the list in constant time as we consider each neighbor, we maintain a set of external indices to each color on the list. After checking all the neighbors, if the list is non-empty, the head of the list points to the color that meets the balanced coloring criterion. This allows us to compute S in $O(b^2)$ time for each node for a distance-2 coloring. Updating the sorted list after coloring a node takes $O(b^2 + p)$ time. Thus the overall time complexity of doing a distance-2 balanced coloring is $O(n(b^2 + p))$. We note that for finite element problems we generally have small b and p and very large n.

7   Results

We consider two applications for our numerical experiments, The first application models a leaky lid-driven cavity using the Navier-Stokes equations. For this applications we use the MATLAB software of [13]. This particular problem has $A \neq A^T$, $B = C$ and $D \neq 0$ in the notation of (1). We use a 16×16 grid with viscosity parameter $\nu = 0.1$ and stabilization parameter $\beta = 0.25$. After removing the constant pressure mode, our system has 705 unknowns. For the splitting of the (1,1) block, $A = F - E$, we use one multi-grid V-cycle with three SOR-Jacobi pre- and post-smoothing steps and relaxation parameter $\omega = 0.25$.

The second application uses the modified Hart's model [15] to model elastic, plastic, anelastic, micro-plastic and micro-anelastic strain and their effects on the permanent deformation of bent beams [30]. The linear system has 6422 unknowns and arises from a Newton iteration to solve the nonlinear problem at each timestep. For this problem, we study the following structural splittings of the (1,1) block A: the diagonal of A, a banded splitting of A with a semi-bandwidth of four, and the ILU(0) factorization of A.

In addition to demonstrating the effectiveness of our preconditioners with approximate Schur complements generated by structured probing, we use the Navier-Stokes problem to illustrate the superiority of structured probing to classic probing. Specifically, we focus on the role of the sparsity pattern chosen for the approximate matrix. We use the prime divisor method for graph coloring to isolate the role of this chosen sparsity pattern. This allows us to use the same probing vectors for classic probing and for structured probing; so, the only difference between the two methods is in the sparsity pattern that we use for the construction of the approximate matrix. Hence, we are not trying to get the most out of structured probing, but rather demonstrate that even using the same vectors as classic probing, reconstruction based on a better

```
function nzval=sp_probe_given_coloring(func, N, rp, ci, p, d)
  %Input parameters
  %  func - Function such that func(x) = A x
  %  N    - (int) number of columns of the matrix
  %  rp   - (vector of ints) the matrix row pointers of H, CSR style
  %  ci   - (vector of ints) the matrix column indices of H, CSR style
  %  p    - (int) the number of colors used by the graph coloring
  %  d    - (vector of ints) a N-vector containing the color of each vertex
  %Return Value
  %  nzval- (vector of doubles) the matrix non-zero entries of
  %          the output matrix, \tilde{K}, CSR style

  %Local variables
  %  I,J  - (ints) counters
  %  X    - (matrix) Matrix of probing vectors X = [x_1,...,x_p]
  %  W    - (matrix) Matrix of resulting vectors W = [w_1,...,w_p], where W=AX

  % Allocations
  W=zeros(N,p);
  X=zeros(N,p);

  % Generate vectors to be used by probing
  for I=1:N, X(I,d(I))=1; end

  % Do the matvecs
  for I=1:p, W(:,I)=func(X(:,I)); end

  % Matrix assembly - put resulting entries into the right place
  for I=1:N,
    for J=rp(I):rp(I+1)-1,
      nzval(J)=W(I,d(ci(J)));
```

Figure 4. Structured probing pseudo-code: probing with a given graph coloring.

sparsity pattern leads to much better eigenvalue clustering and convergence.

Furthermore, we examine the use of incomplete factorizations (ILU(0)) for the approximate Schur complement matrices generated by structured probing as a means of further reducing the cost of the preconditioners (2) and (5).

For the metal deformation application we use structured probing with the distance-2 balanced coloring algorithm of [8]. We also use ILU(0) to factor the approximate Schur complement generated by structured probing. Here we focus on GMRES convergence and wall-clock time.

7.1 Benefits of Structured Probing

Figures 5(a) and 5(b) show the eigenvalue distributions for the related system (5) for classic probing and structured probing using 13 probing vectors. For scaling purposes, we exclude two negative eigenvalues at approximately $(-73, 0)$ and $(-113, 0)$ for the classic probing case. We use a nine-point stencil on the element connectivity graph to define the sparsity pattern H for structured probing (SP). The prime divisor method described in Section 4, yields a graph coloring requiring 13 vectors. The vectors used for structured probing and classic probing are the same. The only difference between the two methods is in the construction of the approximate matrix.
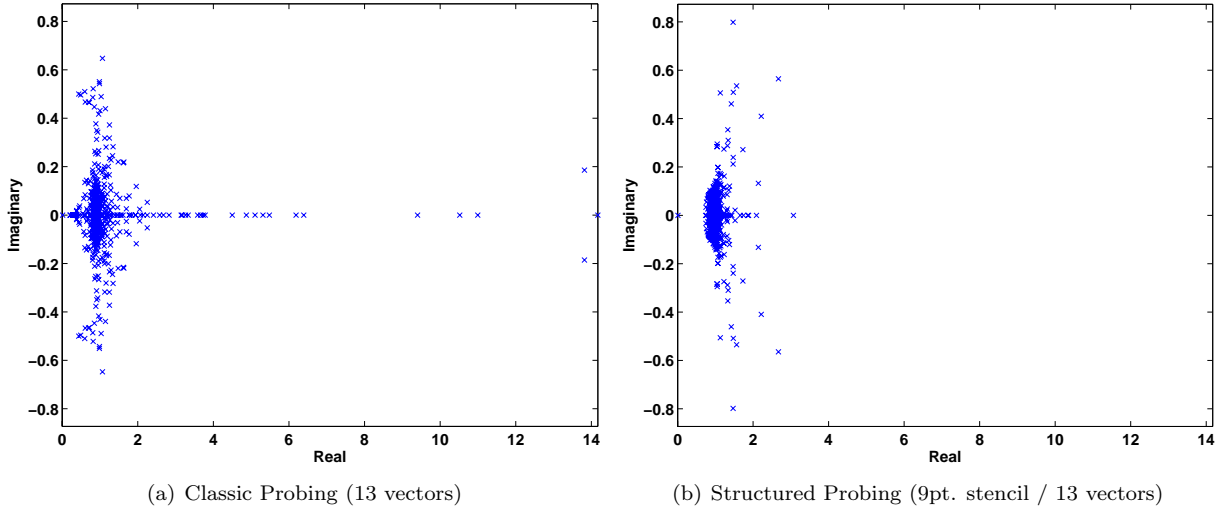

(a) Classic Probing (13 vectors)   (b) Structured Probing (9pt. stencil / 13 vectors)

Figure 5. Eigenvalues for the related system (5) derived from the Navier-Stokes problem with one V-cycle as splitting of the (1,1) block and approximate Schur complements using classic and structured probing with exact factorizations.

Structured probing yields much better clustering than classic probing, especially near the origin. Structured probing has only one small eigenvalue (about 0.01); the others are well separated from zero. Classic probing has many eigenvalues clustering near the origin. This leads to worse convergence behavior; see Figure 7(a).

We see similar results for the eigenvalues of the block-diagonally preconditioned system (2) for classic and structured probing; see Figures 6(a) and 6(b). Both probing and structured probing have one small eigenvalue (about 0.01), but structured probing clusters eigenvalues much further away from the origin.

Figures 7(b) and 7(a) show the convergence of GMRES for the preconditioned systems. The difference between classic and structured probing is quite pronounced. For both preconditioned systems, structured probing from a five-point stencil using seven vectors has a lower iteration count than classic probing even with thirteen vectors. We also note that using the related system (5) leads to significantly faster convergence than using the block-diagonally preconditioned system (2) for all probing variants.

Figures 8(a) and 8(b) show the eigenvalues for the related system and block-diagonally preconditioned system for structured probing with both five-point (seven vector) and nine-point (thirteen vector) stencils. Note that, barring a few outliers, for both kinds of preconditioners the eigenvalue clustering is significantly better for the nine-point stencil, especially near the origin. Krylov-subspace methods tend to find and
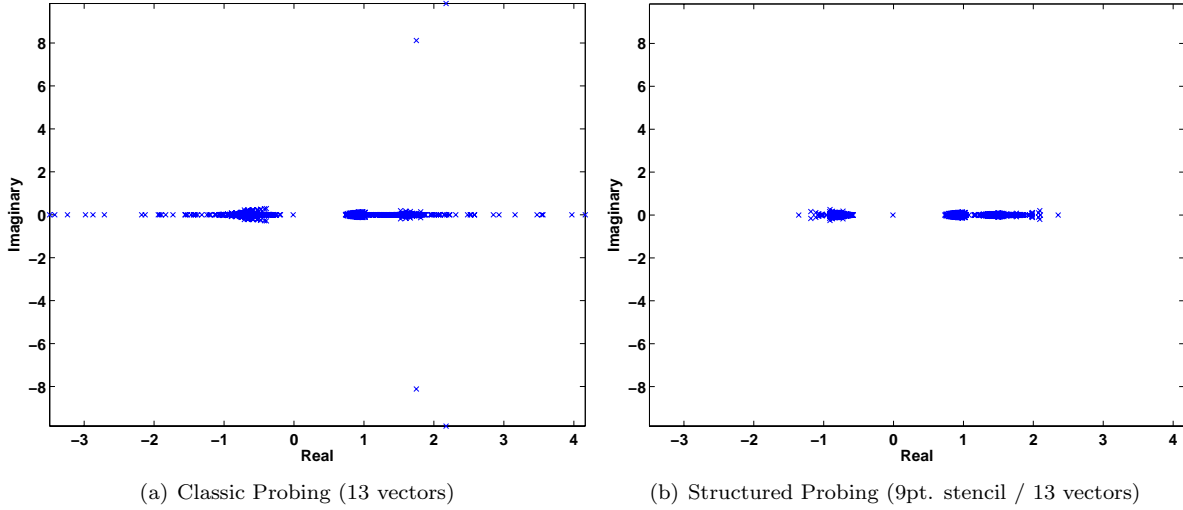
(a) Classic Probing (13 vectors)　　　　(b) Structured Probing (9pt. stencil / 13 vectors)

Figure 6. Eigenvalues for the block-diagonally preconditioned system (2) derived from the Navier-Stokes problem with one V-cycle as splitting of the (1,1) block and approximate Schur complements using classic and structured probing with exact factorizations.
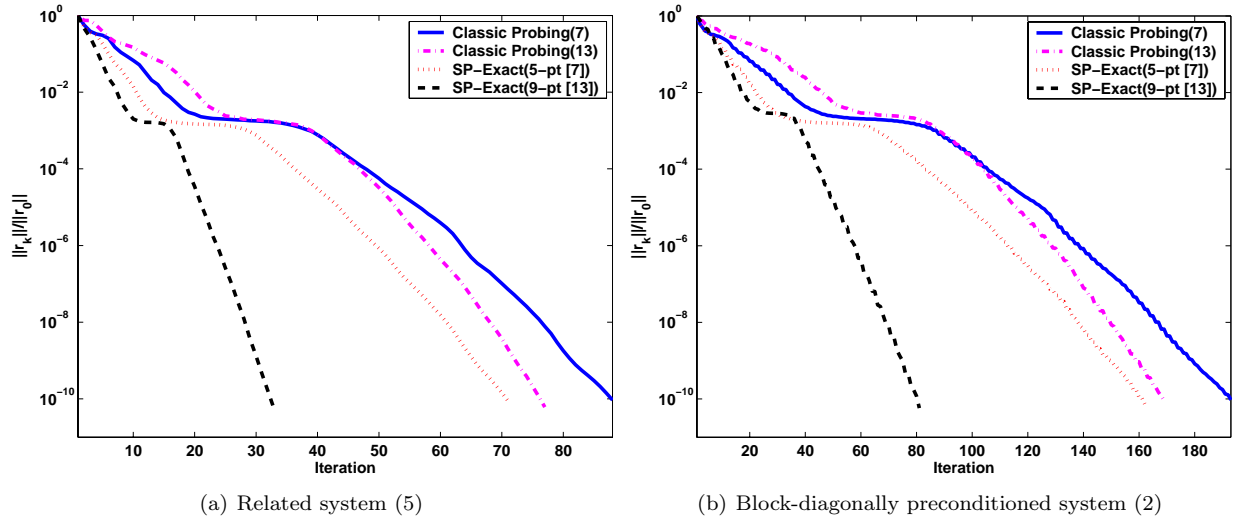

(a) Related system (5)　　　　(b) Block-diagonally preconditioned system (2)

Figure 7. GMRES convergence for the Navier-Stokes problem with one V-cycle as splitting of the (1,1) block and approximate Schur complements using classic and structured probing with exact factorizations.

"remove" outlying eigenvalues quickly. Therefore, these eigenvalues do not affect the convergence rate after some number of initial iterations. Thus, the significantly better eigenvalue clustering obtained using the nine-point stencil leads to a significantly improved convergence rate for GMRES.
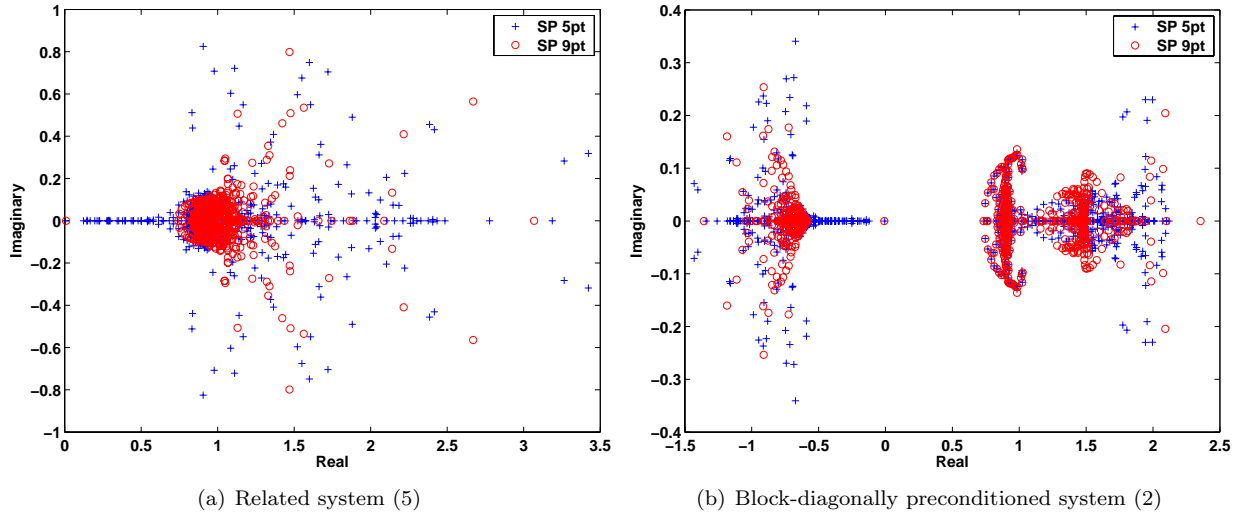
(a) Related system (5) (b) Block-diagonally preconditioned system (2)

Figure 8. Eigenvalues for the related system (5) and the block-diagonally preconditioned system (2) derived from the Navier-Stokes problem with one V-cycle as splitting of the (1,1) block and approximate Schur complements using structured probing with exact factorizations.

7.2 Employing ILU(0) for the Approximate Schur Complement from Structured Probing

The benefits of structured probing come at the cost of having an approximate Schur complement that is more expensive to factor. For instance, if the chosen sparsity pattern looks similar to an n-dimensional Laplacian (because A and hence F are related to a Laplacian), the large bandwidth can yield significant fill-in, making the exact factorization of the approximate Schur complement matrix expensive to compute and apply. Therefore, we use an inexact factorization of the approximate Schur complement to define S_2^{-1}. In practice, this leads to a negligible deterioration in convergence while reducing the overhead of applying structured probing significantly. We use an ILU(0) factorization for this problem. For symmetric problems an IC(0) factorization should be used. Since ILU(0) and IC(0) have linear cost in the number of unknowns, the overall cost remains $O(m)$.

Figure 9 shows the eigenvalue distributions for both preconditioned systems with structured probing using a nine point stencil (13 vectors) for both the exact and ILU(0) factorizations of the approximate Schur complement. Figure 10 shows the convergence results for both preconditioned systems, using structured probing with 9 and with 13 vectors. Using ILU(0) instead of an exact factorization changes the eigenvalue distribution slightly, and leaves the clustering is essentially equivalent. The impact of such a change on the convergence behavior is negligible. Given the significant difference in cost between exact and inexact factorizations, using ILU(0) is more cost-effective than an exact factorization.

7.3 Computational Results for Metal Deformation

The metal deformation problem arises from a finite element mesh, where the second set of variables corresponds to nodes in the center of the elements. For structured probing, we approximate the Schur complement
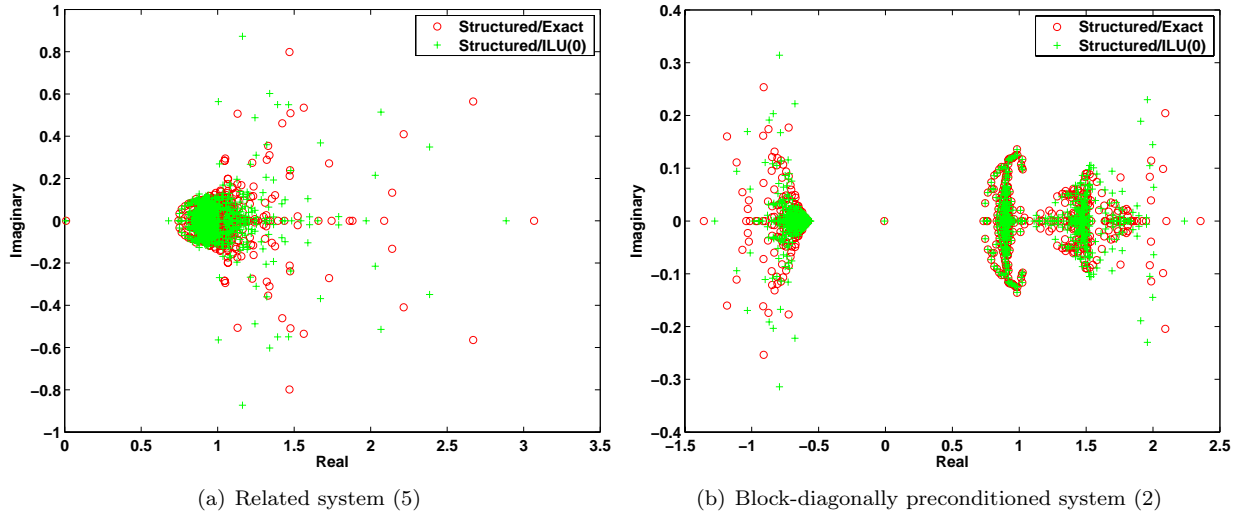
13

(a) Related system (5)   (b) Block-diagonally preconditioned system (2)

Figure 9. Eigenvalues for the related system (5) and the block-diagonally preconditioned system (2) derived from the Navier-Stokes problem with one V-cycle as splitting of the (1,1) block and approximate Schur complements using structured probing (13 vectors) with exact and ILU(0) factorizations.
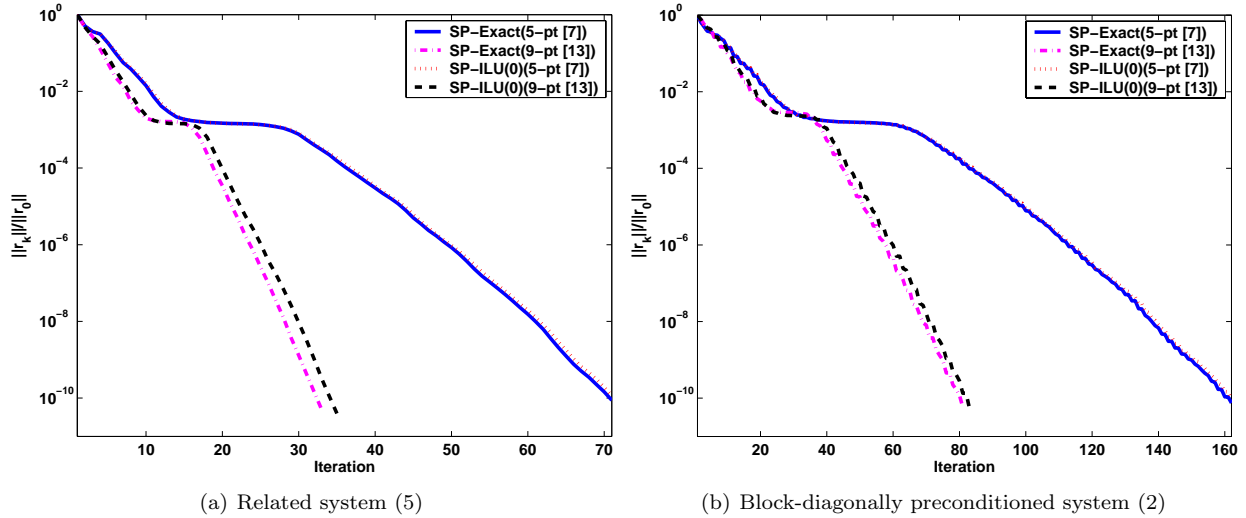

(a) Related system (5)   (b) Block-diagonally preconditioned system (2)

Figure 10. GMRES convergence for the Navier-Stokes problem with one V-cycle as the splitting of the (1,1) block and an inexact Schur complement computing using structured probing, using both exact and ILU(0) factorizations.

with a matrix that has the sparsity pattern of the element-element connectivity graph of the original problem (i.e. G_1 in subsection 4.1). Using a distance-2 balanced coloring, we can build our approximation using only nine probing vectors.

14

(a) GMRES Convergence for ILU(0) Splitting   (b) Wall Clock Time
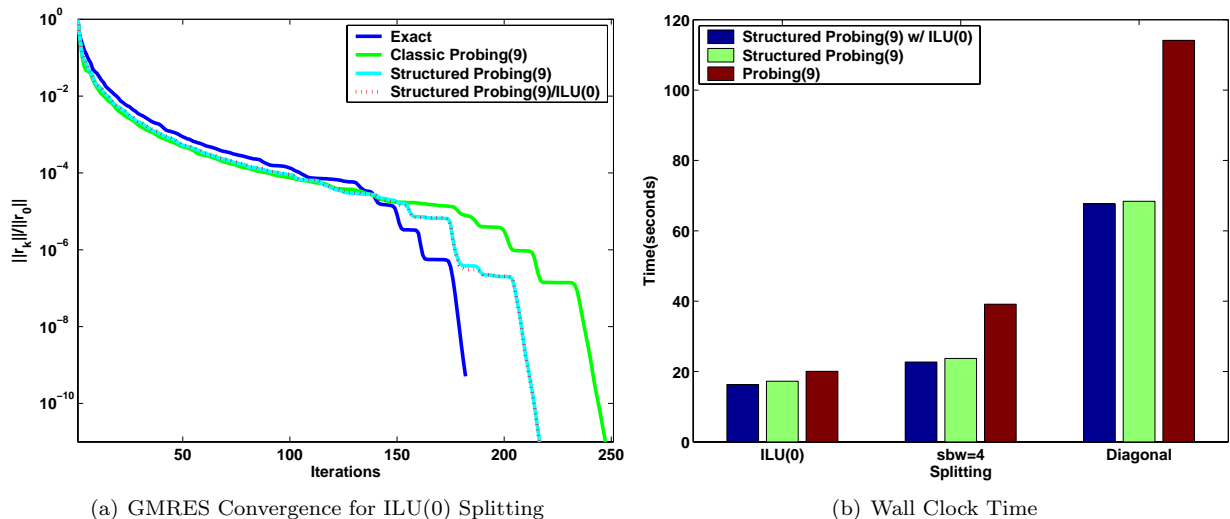
Figure 11. GMRES convergence and wall clock time for the metal deformation problem, for various probing-based inexact Schur complements in the related system (5) for three different splittings of the (1,1) block (ILU(0), banded matrix with semibandwith 4, and diagonal).

We present GMRES convergence results and wall clock time for a single linear system from the metal deformation problem in Figures 11(a) and 11(b), respectively. For the convergence results, we use an ILU(0) splitting of the (1,1) block, A, and we compare the convergence for the exact Schur complement, an approximate Schur complement using classic probing, and an approximate Schur complement using structured probing with both exact and ILU(0) factorizations. With respect to the choice of approximate Schur complement, Figure 11(a) shows that structured probing leads to faster convergence than classic probing. In terms of execution time, Figure 11(b) shows structured probing leads to a savings in time of 15% to 40% over classic probing, with the inexact factorization saving an additional 5% or so of execution time.

It should be noted that this problem models a long, thin, piece of metal. So, although the problem is three dimensional, it is similar in nature to a one-dimensional problem (the elements are also ordered appropriately). Therefore, classic probing for the Schur complement does very well, as this problem shows the same type of 1-D decay as the Schur complements for 2-D domain decomposition problems for which classic probing was designed. This is a relatively easy 3-D problem for classic probing and the improvements due to structured probing should be viewed in that light. Our results show not only the efficacy of structured probing, but also the potential benefit of the use of inexact factorizations for the approximations from structured probing.

8   Conclusions and Future Work

We have shown that classic probing [4], although it was designed for 2-D domain decomposition problems, can be generalized to reconstruct exactly matrices of arbitrary sparsity structure or approximate matrices that have a suitable decay property relative to a chosen sparsity structure. This makes probing a very powerful technique in preconditioning saddle-point problems, if a good estimate of the probing pattern can

be made *a priori*. The results presented in this paper show the effectiveness of these preconditioners using structured probing in terms of eigenvalue clustering, rate of convergence and execution time.

As future work, we seek to develop estimates for $\|\mathcal{E}\|$ for structured probing methods. We also seek to dynamically adapt the *a priori* chosen sparsity structure for structured probing. Furthermore, we plan to identify additional problems that yield decay properties that can be exploited by structured probing, especially based on more algebraic criteria. We will also work on developing an accurate eigenvalue analysis for systems preconditioned using structured probing. Finally, we intend to look at schemes for updating and reusing (generalized) saddle point preconditioners with probing-based inexact Schur complements.

References

[1] C. Bernardi, C. Canuto, and Y. Maday. Generalized inf-sup conditions for Chebyshev spectral approximation of the Stokes problem. *SIAM J. on Numer. Anal.*, 25(6):1237–1271, 1988.

[2] D. Braess. *Finite Elements: Theory, fast solvers and applications in solid mechanics.* Cambridge University Press, 2nd edition, 2001.

[3] J.H. Bramble and J.E. Pasciak. A preconditioning technique for indefinite systems resulting from mixed approximations of elliptic problems. *Mathematics of Computation*, 50(181):1–17, January 1988.

[4] T.F. Chan and T.P. Mathew. The interface probing technique in domain decomposition. *SIAM J. Matrix Anal. Appl.*, 13(1):212–238, January 1992.

[5] Edmond Chow. A priori sparsity patterns for parallel sparse approximate inverse preconditioners. *SIAM J. Sci. Comput.*, 21(5):1804–1822, 2000.

[6] T.F. Coleman and J.J. Moré. Estimation of sparse Jacobian matrices and graph coloring problems. *SIAM J. Numer. Anal.*, 20(1):287–209, 1983.

[7] T.F. Coleman and J.J. Moré. Estimation of sparse Hessian matrices and graph coloring problems. *Math. Programming*, 28:243–270, 1984.

[8] J. Cullum and M. Tuma. Matrix-free preconditioning using partial matrix estimation. Technical Report 898, Institute of Computer Science, Academy of Sciences of the Czech Republic, April 2004.

[9] A.R. Curtis, M.J.D. Powell, and J.K. Reid. On the estimation of sparse Jacobian matrices. *J. Inst. Math. Appl.*, 13:117–119, 1974.

[10] E. de Sturler and J. Liesen. Block-diagonal and constraint preconditioners for nonsymmetric indefinite linear systems. Part I: Theory. Technical Report 36-2003, Institute of Mathematics, Technical University of Berlin, September 2003. Accepted for publication in SIAM J. Sci. Comput.

[11] H.C. Elman. Preconditioning for the steady-state Navier-Stokes equations with low viscosity. *SIAM J. Sci. Comput.*, 20(4):1299–1316, 1999.

[12] H.C. Elman and D. Silvester. Fast nonsymmetric iterations and preconditioning for Navier-Stokes equations. *SIAM J. Sci. Comput.*, 17:33–46, January 1996.

[13] H.C. Elman, D.J. Silvester, and A.J. Wathen. Iterative methods for problems in computational fluid dynamics. In *Winter School on Iterative Methods in Scientific Computing and Applications*. Chinese University of Hong Kong, 1996.

[14] H.C. Elman, D.J. Silvester, and A.J. Wathen. Performance and analysis of saddle point preconditioners for the discrete steady-state Navier-Stokes equations. *Numer. Math.*, pages 665–688, 2002.

[15] H. Garmestani, M.R. Vaghar, and E.W. Hart. A unified model for inelastic deformation of polycrystalline materials — application to transient behavior in cyclic loading and relaxation. *International Journal of Plasticity*, pages 1367–1391, 2001.

[16] A.H. Gebremedhin, F. Manne, and A. Pothen. What color is your Jacobian? Graph coloring for computing derivatives. To appear in SIAM Review.

[17] A.H. Gebremedhin, F. Manne, and A. Pothen. Graph coloring in optimization revisited. Technical Report 226, Department of Informatics, University of Bergen, January 2002.

[18] L. Giraud and R.S. Tuminaro. Schur complement preconditioners for anisotropic problems. *IMA J. Numerical Analysis*, 19:1–17, 1999.

[19] I.C.F. Ipsen. A note on preconditioning nonsymmetric matrices. *SIAM J. Sci. Comput.*, 23(3):1050–1051, 2001.

[20] D. Loghin and A.J. Wathen. Schur complement preconditioners for elliptic systems of partial differential equations. *Numer. Linear Algebra Appl.*, 10:423–443, 2003.

[21] S.T. McCormick. Optimal approximation of sparse Hessians and its equivalence to a graph coloring problem. *Math. Programming*, 26:153–171, 1983.

[22] M.F. Murphy, G.H. Golub, and A.J. Wathen. A note on preconditioning for indefinite linear systems. *SIAM J. Sci. Comput.*, 21(6):2969–1972, 2000.

[23] R. Nicolaides. Existence, uniqueness and approximation for generalized saddle point problems. *SIAM Journal on Numerical Analysis*, 19(2):349–357, 1982.

[24] I. Perugia and V. Simoncini. Block-diagonal and indefinite symmetric preconditioners for mixed finite element formulations. *Numer. Linear Algebra Appl.*, 7:585–616, 2000.

[25] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations*. Springer-Verlag, 2nd edition, 1997.

[26] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, 2nd edition, 2003.

[27] C. Siefert. *Structured Probing Toolkit*, 2005. `http://www.cse.uiuc.edu/~siefert/structured_probing/`.

[28] C. Siefert and E. de Sturler. Preconditioners for generalized saddle-point problems. Technical Report UIUCDCS-R-2004-2448, Department of Computer Science, University of Illinois at Urbana-Champaign, June 2004.

[29] J.C. Tannehill, D.A. Anderson, and R.H. Pletcher. *Computational Fluid Mechanics and Heat Transfer*. Taylor & Francis, Philiadelphia, 2nd edition, 1997.

[30] L. Zhu, A.J. Beaudoin, and S.R. MacEwan. A study of kinetics in stress relaxation of AA 5182. In *Proceedings of TMS Fall 2001: Microstructural Modeling and Prediction During Thermomechanical Processing*, pages 189–199, 2001.